

In the Claims:

Please cancel claims 20-25. Please amend claims 1 and 3-19. Please add new claims 26-39. The claims are as follows.

1. (Currently amended) A system comprising multiple processing servers and a management server for managing the multiple processing servers, each of which ~~executes one or more programs~~ processing server adapted to execute a program assigned thereto from among multiple programs for processing one or more steps included in performing a requested job, and a management server for managing the multiple processing servers;
the management server comprising:

an execution direction generating unit for generating an execution direction, said execution direction including identification information identifying each of the multiple programs, input and/or output files for each of the multiple programs, and the an execution order of the multiple programs~~[[;]]~~ for performing the requested job, said execution order identifying a first program of the multiple programs to be executed before any other program of the multiple programs is executed;

an input data sending unit for sending~~[[,]]~~ input data and the execution direction to a first processing server of the multiple processing servers for executing ~~[[a]]~~ the first program ~~to be executed first~~ with the input data ~~to be inputted in for the job as input, the~~ input data and the execution direction in association with each other;

a processing server information storing unit for storing identification information identifying each program of the multiple programs and identification information on ~~[[a]]~~

each processing server for executing the program assigned to each processing server in association with each other; and

an inquiry responding unit for acquiring identification information on a processing server corresponding to identification information on a program to be executed next from the processing server information storing unit [[to send]] and for sending the identification information as a reply to the inquiry; to an inquiring processing server of the multiple processing servers in response to an inquiry for said identification information received from the inquiring processing server after the inquiring processing server has completed execution of the program assigned thereto, said identification information identifying a next processing server to execute a next program included in the execution direction; and

each processing server of the multiple processing servers comprising:

a program executing unit for executing the assigned program with the using received input data as input to update and for updating the input data following said executing the assigned program;

a processing server selecting unit for sending, to the management server, an inquiry for identification information identifying a next processing server to execute the next program included in the execution direction after said each processing server has completed execution of the program assigned thereto about a processing server for executing a program to be executed next with the updated input data as input, based on the execution direction received in association with the input data to select a processing server to be identified according to identification information as a result of the inquiry;

and

an input data transferring unit for sending, to the next processing server identified in the identification information received ~~selected~~ by the processing server selecting unit from the management server in response to said inquiry, the received execution direction and the updated input data ~~in association with each other~~.

2. (Canceled)

3. (Currently amended) The system ~~according to Claim 2 of claim 1,~~

wherein each processing server of the multiple processing servers further comprises a caching unit for caching said identification information of ~~[[a]]~~ said next processing server which the processing server selecting unit of said each processing server has acquired as a result of said inquiry ~~of the inquiry responding unit in the past, in association with identification information on a program to be executed by the processing server; and~~

wherein the identification information is cached by the caching unit; and

wherein the processing server selecting unit selects, ~~on condition that identification information of a program to be executed is cached by the caching unit, a processing server corresponding to the~~ next processing server to execute the next program.

4. (Currently amended) The system ~~according to Claim 2 of claim 1,~~

wherein in response to receiving the input data, the program executing unit sends a receiving notification indicating that input data has been received to the management server, and

in response to terminating execution of the assigned program, the program executing unit sends an termination notification indicating that execution of the assigned program has been terminated;

wherein the processing server information storing unit stores executability information based on the receiving notification and the termination notification on whether or not it is possible for each processing server of the multiple processing servers to newly receive input data and execute ~~[[a]]~~ its assigned program; and

~~wherein the inquiry responding unit selects, from among multiple processing servers for executing a program for which an inquiry has been made, a processing server capable of newly receiving input data and executing the program based on the executability information to send identification information of the selected processing server~~ the next processing server to execute the next program.

5. (Currently amended) The system ~~according to Claim of claim~~ of claim 4,

wherein the management server further comprises a program activating unit for, on condition that ~~processing servers~~ more than a predetermined reference rate ~~among the multiple percentage of the~~ processing servers for ~~executing the program for which the inquiry responding unit has received an inquiry,~~ are not capable of newly receiving input data and executing the next program, activating the next program on ~~any of one~~ one processing server~~[[s]]~~ which ~~have~~ has not activated the next program yet; and

wherein the inquiry responding unit sends identification information of the one processing server on which the next program has been activated by the program activating unit.

6. (Currently amended) The system ~~according to Claim~~ of claim 1,

wherein each processing server of the multiple processing servers comprises:

a history storing unit for storing history of the input data and the execution direction sent by the input data transferring unit to the processing server; and

a fault occurrence determining unit for determining whether or not any fault has occurred in execution of a program on a sending-destination processing server to which the input data transferring unit has sent the updated input data and the execution direction; ~~and~~

wherein in response to determining ~~that a fault has occurred~~ by the fault occurrence determining unit that said any fault has occurred, the processing server selecting unit selects a different processing server for executing ~~[[a]]~~ the next program to be executed next; and

wherein the input data transferring unit acquires the input data and the execution direction from the history storing unit to send the input data and the execution direction to the different processing server selected by the processing server selecting unit.

7. (Currently amended) The system ~~according to Claim~~ of claim 6,

wherein in response to receiving the input data, the program executing unit sends a receiving notification indicating that input data has been received ~~[[to]]~~ from the management server,

wherein in response to terminating execution of the assigned program, the program executing unit sends an termination notification to the management server indicating that execution of the assigned program has been terminated; and

wherein the management server further comprises a fault occurrence notification sending unit for, if the receiving notification is received from any processing server and, after that, the termination notification is not received within a predetermined reference processing time from ~~the~~ said any processing server, sending a fault occurrence notification indicating that a fault has occurred in execution of a program on ~~the~~ said any processing server to a sending-source processing server which has sent the input data to ~~the~~ said any processing server; and

wherein the fault occurrence determining unit determines, if receiving the fault occurrence notification, that a fault has occurred in execution of a program on a sending-destination processing server to which the input data transferring unit has sent the updated input data and the execution direction.

8. (Currently amended) The system ~~according to Claim~~ of claim 6,

wherein in response to receiving the input data and execution direction by a sending-destination processing server, on ~~condition that newly receiving input data and~~ determining that it is impossible to execute a program, the program executing unit of the sending-destination processing server sends a refusal notification indicating refusal of the input data to a sending-source processing server which has sent the input data and the execution direction to the sending-destination processing server; and

wherein in response to receiving the refusal notification from the sending-destination processing server, the fault occurrence determining unit of the sending-source processing server determines that a fault has occurred in execution of a program on the sending-destination processing server.

9. (Currently amended) The system according to Claim of claim 6, wherein the management server further comprises a deletion directing unit for, on condition that the job is completed by execution of the multiple programs, causing the input data and the execution direction to be deleted from the history storing section of each processing server of the multiple processing servers

10. (Currently amended) The system according to Claim of claim 6, wherein the processing server selecting unit of each processing server of the multiple processing servers selects, from among multiple processing servers activating a program to be executed next with the updated input data, a processing server which communicates with the processing server with a higher communication speed as the different processing server in preference to a server with a lower communication speed.

11. (Currently amended) The system according to Claim of claim 1, wherein each processing server of the multiple processing servers further comprises:

a history storing unit for storing history information on data on the processing server which the program executing unit has changed by executing the program, in association with information which enables restoration to the state before the program executing unit has changed; and

a change restoring unit for, if a fault has occurred in execution of a program on a sending-destination processing server to which the input data transferring unit has sent the updated input data and execution direction, restoring data changed by the program executing unit to the original

state of the changed data based on the history information in the history storing unit.

12. (Currently amended) The system ~~according to Claim~~ of claim 1, wherein the management server further comprises a program activating unit for detecting each of programs to be executed for and after the second time ~~with using the input data to be inputted in the job as input~~, based on the ~~generated~~ execution direction, and activating each of the detected programs on any processing server different from the sending-destination server of the input data sending unit.

13. (Currently amended) The system ~~according to Claim~~ of claim 12,

wherein the program executing unit of each processing server of the multiple processing servers notifies the management server of throughput required for execution of a program in the past; and

wherein the program activating unit of the management server activates, on condition that the past throughput of a program notified by the program executing unit exceeds the maximum throughput to be processed by a processing server already activating the program, the program on any processing server different from the sending-destination server of the input data sending unit.

14. (Currently amended) The system ~~according to Claim~~ of claim 1, wherein the management server further comprises a program activating unit for detecting each of programs to be executed ~~with using the input data to be inputted in the job as input~~; and, for each of the detected programs, on condition that throughput required for the program to be executed with the input data exceeds the maximum throughput to be processed by ~~that~~ a processing server already

activating the program, activates the program on any processing server different from the sending-destination server of the input data sending unit.

15. (Currently amended) The system ~~according to Claim~~ of claim 1, wherein on condition that the program executing unit does not receive the input data and execution direction within a predetermined reference waiting time after receiving input data and execution direction ~~last~~, the program executing unit stops execution of a program by the processing server that is executing the program.

16. (Currently amended) The system ~~according to Claim~~ of claim 1, wherein the management server further comprises a processing server changing unit for, on condition that the usage rate of computation resources to be used for execution of a program by the program executing unit is below a predetermined reference usage rate, causing a different processing server with a less maximum throughput than that of the processing server to execute the program.

17. (Currently amended) The system ~~according to Claim~~ of claim 1, wherein in response to receiving input data and execution direction by a sending-destination processing server, the program executing unit detects that there still exists a program which should have been already executed based on received execution direction, and executes the detected program on condition that it is possible for the sending-source processing server to execute the program.

18. (Currently amended) The system ~~according to Claim~~ of claim 1,

wherein the input data transferring unit of each processing server of the multiple processing servers creates a digital signature of the updated input data or the execution information to send the created digital signature in association with the input data and execution direction to be covered by the digital ~~sign~~ signature; and

wherein the program executing unit of a processing server executes a program on condition that the digital signature is correctly verified.

19. (Currently amended) A management server for managing multiple processing servers, each of ~~which executes one or more programs~~ processing server adapted to execute a program assigned thereto from among multiple programs for ~~processing one or more steps included in performing a~~ requested job[[;]], ~~each of the multiple processing servers comprising:~~

~~a program executing unit for executing the assigned program with the received input data as input to update the input data;~~

~~a processing server selecting unit for sending, to the management server, an inquiry about a different processing server for executing a program to be executed next with the updated input data as input, based on the execution direction received in association with the input data to select the processing server to be identified according to identification information as a result of the inquiry; and~~

~~an input data transferring unit for sending, to the processing server selected by the processing server selecting unit, the received execution direction and the updated input data in association with each other; and~~

the management server comprising:

~~an execution direction generating unit for generating an execution direction, said~~ execution direction ~~including identification information identifying each of the multiple programs, input and/or output files for each of the multiple programs, and the an~~ execution order of the multiple programs[[;]] for performing the requested job, said execution order identifying a first program of the multiple programs to be executed before any other program of the multiple programs is executed;

~~an input data sending unit for sending[[,]]~~ input data and the execution direction

to a first processing server of the multiple processing servers for executing [[a]] the first program to be executed first with the input data to be inputted in for the job as input, the input data and the execution direction in association with each other;

a processing server information storing unit for storing identification information identifying each program of the multiple programs and identification information on [[a]] each processing server for executing the program assigned to each processing server in association with each other; and

an inquiry responding unit for acquiring identification information on a processing server corresponding to identification information on a program to be executed next from the processing server information storing unit [[to send]] and for sending the identification information as a reply to the inquiry, to an inquiring processing server of the multiple processing servers in response to an inquiry for said identification information received from the inquiring processing server after the inquiring processing server has completed execution of the program assigned thereto, said identification information identifying a next processing server to execute a next program included in the execution direction.

20-25. (Canceled)

26. (New) A method for performing a requested job by a system that comprises multiple processing servers and a management server for managing the multiple processing servers, each processing server adapted to execute a program assigned thereto from among multiple programs for performing the requested job, said method comprising:

generating, by the management server, an execution direction, said execution direction including identification information identifying each of the multiple programs, input and/or output files for each of the multiple programs, and an execution order of the multiple programs for performing the requested job, said execution order identifying a first program of the multiple programs to be executed before any other program of the multiple programs is executed;

sending, by the management server to a first processing server of the multiple processing servers, the execution direction and input data for subsequent execution of the first program;

executing, by the first processing server, the first program using the input data as input, wherein said executing the first program results in said input data being updated;

after said executing the first program, sending, by the first processing server to the management server, an inquiry for identification information that identifies a second processing server to execute a second program included in the execution direction;

after sending the inquiry, receiving, by the first processing server from the management server, the identification information; and

sending, by the first processing server to the second processing server, the execution direction and the updated input data for subsequent execution of the second program.

27. (New) The method of claim 26, further comprising after said receiving the execution

direction and input data:

sending, by the first processing server to the management server, a receiving notification indicating that the first processing server has received the execution direction and input data sent by the management server.

28. (New) The method of claim 26, further comprising after completion of said executing the first program:

sending, by the first processing server to the management server, a termination notification indicating that the first processing server has completed execution of the first program.

29. (New) The method of claim 26, wherein after said sending, by the first processing server to the second processing server, the execution direction and the updated input data for subsequent execution of the second program, the second processing server is designated as a current processing server, the second program is designated as a current program, and the updated input data is designated as current input data, and the method further comprises executing a loop that comprises at least one iteration such that executing the loop comprises performing each iteration of the at least one iteration, wherein each iteration when performed is designated as a current iteration, and wherein performing each iteration of the loop comprises:

if the current processing server is unable to execute the current program then not executing the current program by the current processing server, otherwise initiating execution of the current program by the current processing server using the current input data as input which

results either in completion of executing the current program by the current processing server with the current input data being updated or an occurrence of a fault during said executing the current program by the current processing server;

if said execution of the current program by the current processing server is initiated and completed, then after completion of executing the current program by the current processing server: sending, by the current processing server to the management server, either a completion notification to indicate that performance of the requested job has been completed or an inquiry for identification information that identifies a next processing server to execute a next program included in the execution direction;

if the inquiry has been sent then after said sending the inquiry and receipt thereof by the management server:

sending, by the management server to the current processing server, the identification information; and

sending, by the current processing server to the next processing server, the execution direction and the updated input data for subsequent execution of the next program to end the current iteration, wherein for the next iteration is to be performed: the next processing server is designated as the current processing server, the next program is designated as the current program, and the updated input data is designated as the current input data.

30. (New) The method of claim 29, wherein during an individual iteration of the at least one iteration said execution of the current program by the current processing server is initiated and

completed.

31. (New) The method of claim 30, wherein the individual iteration is the last iteration of the at least one iteration and during said last iteration after said completion of said executing the current program by the current processing server, said sending the completion notification by the current processing server to the management server is performed.

32. (New) The method of claim 30, wherein during the individual iteration after said completion of said executing the current program by the current processing server is performed, said sending the inquiry by the current processing server to the management server is performed.

33. (New) The method of claim 29, wherein during an individual iteration of the at least one iteration either the current processing server is unable to execute the current program or said execution of the current program by the current processing server is initiated such that said fault occurs during said executing the current program by the current processing server, and wherein the method further comprises during said individual iteration:

sending, by the current processing server to a prior processing server that had sent the execution direction and the updated input data to the current processing server, a refusal notification indicating refusal of the updated input data;

sending, by the prior processing server to the management server, an inquiry for different identification information that identifies a different processing server to execute the next program included in the execution direction;

sending, by the management server to the prior processing server, the different identification information; and

sending, by the prior processing server to the different processing server, the execution direction and the updated input data for subsequent execution of the next program during the individual iteration such that the different processing server is designated as the current processing server during the individual iteration.

34. (New) The method of claim 33, wherein during the individual iteration, the current processing server is unable to execute the current program.

35. (New) The method of claim 33, wherein during the individual iteration, said execution of the current program by the current processing server is initiated such that said fault occurs during said executing the current program by the current processing server.

36. (New) The method of claim 29, wherein during an individual iteration of the at least one iteration said execution of the current program by the current processing server is initiated such that said fault occurs during said executing the current program by the current processing server, wherein the current processing server is unable to send a refusal notification to a prior processing server that had sent the execution direction and the updated input data to the current processing server, wherein the refusal notification if sent would have indicated refusal of the updated input data, and wherein the method further comprises during said individual iteration:

determining, by the management server, that said fault has occurred;

sending, by the management server to the prior processing server, a fault occurrence notification indicating that said fault has occurred;

selecting, by the management server a different processing server to execute the next program included in the execution direction; and

sending, by the prior processing server to the different processing server, the execution direction and the updated input data for subsequent execution of the next program during the individual iteration such that the different processing server is designated as the current processing server during the individual iteration.

37. (New) The method of claim 29, wherein during an individual iteration of the at least one iteration said execution of the current program by the current processing server is initiated such that said fault occurs during said executing the current program by the current processing server, and wherein the method further comprises during said individual iteration:

determining, by the management server, that said fault has occurred and that the requested job cannot be performed by avoiding said fault; and

sending, by the management server to all prior servers that have performed a prior execution of a program of the multiple programs during any prior iteration of the at least one iteration, a restoration directive to restore all data changed during said prior execution by said all prior servers.

38. (New) A system comprising a plurality of processing servers and a managing server for managing the multiple processing servers, said system adapted to perform the method of claim

26, wherein the plurality of processing servers consists of the multiple processing servers, and wherein the managing server consists of the management server.

39. (New) Storage media, comprising a computer readable control programs embodied therein, said control programs adapted to be executed by a managing server and a plurality of processing servers to perform the method of claim 26, wherein the plurality of processing servers consists of the multiple processing servers, and wherein the managing server consists of the management server.